ECS 174 & ECS 289L
University of California, Davis
Spring 2024, Pirsiavash

Isolating Vocals from Mixed Audio

Austin Shih and Brandon Wong
We pair programmed for the entire project, contributing equally.

1 Abstract

Decomposing a song's audio signal into vocals and the instrument backing can be done similarly to image segmentation tasks. Audio signals are transformed into spectrograms, which we then extract a vocal spectrogram from. Using methodologies from a prior research paper, *Singing voice separation with deep U-Net convolutional networks*[1], we implement a deep U-Net, which has been proven to be effective at reproducing both high-level features and low-level detail. This implementation validates the effectiveness of the U-Net in audio isolation, as well as provides a practical tool for future exploration.

2 Introduction

Splitting songs into multiple tracks based off of instrument is an extremely valuable problem to solve. With an effective model to split songs with, you can perform further analysis on songs, gaining richer information to perform Music Information Retrieval tasks such as mood classification or sentiment analysis. There are also commercial uses for this project, such as song splitting for karaoke machines.

For this project, we focused on isolating vocal tracks from songs by running a deep U-Net architecture on spectrograms of songs. The goal is to input a mixed signal into our model and have the model train a mask that can be multiplied with our original song data to output the original mixed signal but with only the vocals.

3 Methodology

We decided to transform this time series-like problem into an image segmentation problem. Although most audio comes in time series formats, there are methods to convert audio data from the time domain into the frequency domain using the FFT (Fast Fourier Transform). This produces a spectrogram, which is a graph of the intensity of frequencies throughout time.
We calculated spectrograms for both the mixed signal and the isolated vocal signal and aimed to use our U-Net architecture to create a mask which could be multiplied with the mixed spectrogram to ultimately give back the isolated vocals.

For our spectrogram calculations, we needed to balance quality with speed of training. Having too high resolution would make training take too long, so we used the following parameters: hop length = 768, window length = 1024, n_fft = 1024, sampling_rate = 8912.
The result was a spectrogram that captured roughly 11 seconds of audio with dimensions (512, 128).

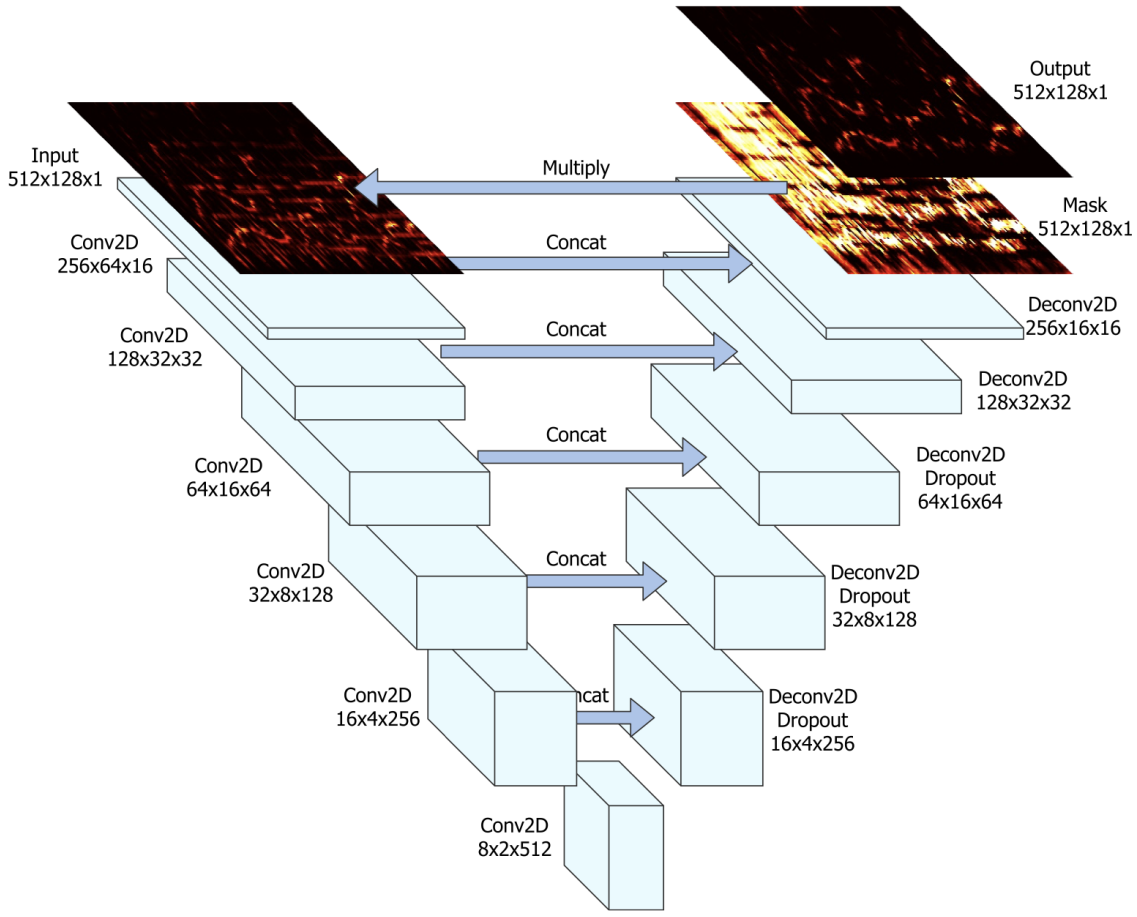

Figure 1: Deep U-Net Architecture

3.1 Architecture

Our network architecture is intended to extract the spectrogram of vocals from the spectrogram of the mixture. We train the model to generate a mask that produces the output spectrogram of vocals when multiplied by the input mixture. We applied the U-Net architecture to this task, given its ability to segment images in fine detail, which is important in audio reconstruction. The U-Net architecture is displayed in Figure 1.

3.1.1 Training

The loss function used to train this model is the L1 loss. This takes the absolute value of the difference between the target spectrogram (vocals) and the masked input spectrogram (result). We used a train-test split of 0.8-0.2. The optimizer used was Adam (adaptive gradient descent with momentum), with a learning rate of 1e-2, decaying by a factor of 0.45 every 10 epochs. Initially, we implemented a fixed learning rate of 1e-3, but we tested with a learning schedule and found that it worked well. We ran the model for 60 epochs and saved checkpoints throughout in order to use the best model for testing.

3.1.2 Architecture Details

Our U-Net implementation is similar to the original research paper. We have a 6 layer network, with a bottleneck layer at the bottom, connecting the encoders to the decoders. Our encoders contain two 2D convolutional layers with a 5x5 kernel and a stride of 2, batch normalization, and leaky ReLU with leakiness 0.2. Each encoder doubles the number of channels, going up to 512 channels in the bottleneck. The bottleneck layer has the same architecture as the encoder, and simply feeds directly into the layer 5 decoder. Our decoders contain a 2D transpose convolutional layer with a 5x5 kernel and a stride of 2, two 2D convolutional layers with a 5x5 kernel and a stride of 2, batch normalization, and leaky ReLU with leakiness 0.2. Each decoder decreases the number of channels by a factor of 2, and ultimately produces a mask of the same size as the input. This decoder architecture is different from the original paper's architecture[2], which did not use any 2D convolutional layers in the decoder. We decided upon this architecture because the original U-Net was implemented in this manner, with the two convolutional layers in the decoders. We also applied a dropout of 0.2 across the entire network, versus the dropout of 0.5 on only the first 3 decoder layers as in the paper. We believed that this extra model complexity would produce a better result.

3.1.3 Vocal Audio Reconstruction

When we generate the spectrogram, we take its absolute value in order to remove complex numbers and allow our model to train on it. In doing so, we lose the phase information, which tells us about the timing of the audio. Replaying the audio without the phase sounds very warbled, so to reconstruct we applied the mask to the original input. This mask only contains magnitude information, which changes the input magnitude while leaving the phase untouched. This reconstruction produces high quality outputs and is audibly effective.

3.2 Dataset

We used the MUSDB18 dataset[4] consisting of 150 full-track songs of different styles. The songs came in Native Instruments stems format (.mp4) format which we used the stempeg library to parse. The dataset can be found here: https://zenodo.org/records/1117372.

4 Evaluation

We evaluate using the mir_eval package[3], which provides us with a method to measure performance. We weren't able to obtain the two datasets that the original paper used because iKala is no longer available and MedleyDB is restricted to authorized users. So, we evaluated on our original dataset from MusDB18. We used the Normalized Signal to Distortion Ratio (NSDR) to evaluate our performance, achieving a score of 6.207. Our NSDR is lower than the paper's performance of 11.094 on iKala and 8.681 on MedleyDB, but we can't necessarily compare against the paper because they used different data. Our NSDR does show that our model truly extracts the vocals, because a positive NSDR means the resulting audio is closer to the reference vocals than the mixed audio. A model achieving an NSDR of 0 means that it does just as badly as simply using the original mixed audio. The paper also calculated the Signal to Interference Ratio (SIR) and Signal to Artifact Ratio (SAR), but these are only applicable when extracting multiple sources of audio (such as instruments). Next, we'll also perform a subjective evaluation. Figure 2 shows some example spectrograms of a clip, and the target and our output appear to be visually similar.

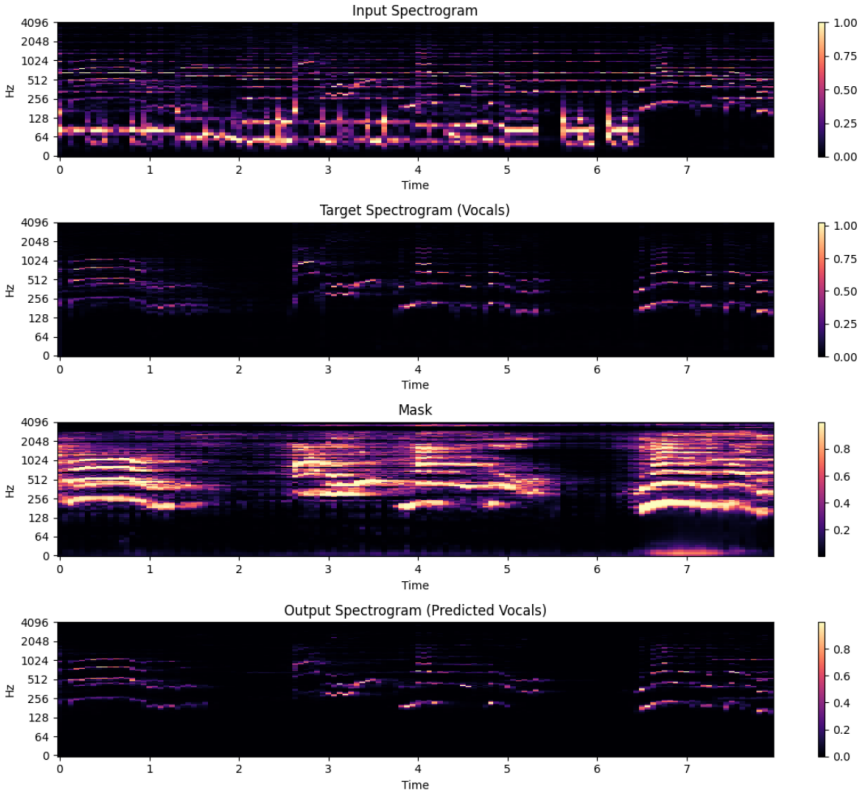

Figure 2: Example Input, Target, Mask, and Output Spectrograms

4.1 Subjective Evaluation

After running our model on several test song clips and also one of our own custom song clips, we noticed that vocals are separated from the rest of the music fairly well. There are often remnants of non-vocal elements, but it is clear the the vocals are the focus. It looks like our mask overshoots values sometimes so the peaks of the vocals may be too high. This could also be the reason that other instruments sometimes are able to leak through since the mask is too forgiving.

5 Conclusion

We explored implementing the U-Net architecture outlined in the original referenced paper, with a few tweaks. This implementation in PyTorch provides a solid foundation for future attempts at isolating vocals from songs. From an objective evaluation with the NSDR and a subjective evaluation of listening to the audio, it's clear that the vocals are extracted from the song. In the future, we may downsample the original audio less, or not at all. This would drastically increase training time, but likely also improve the result. We might also see more success with a larger dataset, if we are able to access iKala or MedleyDB. Subjectively, this model works well, but has opportunity for growth.

6 References

1. Jansson, A., Humphrey, E., Montecchio, N., Bittner, R., Kumar, A. & Weyde, T. (2017). Singing voice separation with deep U-Net convolutional networks. Paper presented at the 18th International Society for Music Information Retrieval Conference, 23-27 Oct 2017, Suzhou, China.

2. Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention.* Springer, 2015.

3. Colin Raffel, Brian McFee, Eric J. Humphrey, Justin Salamon, Oriol Nieto, Dawen Liang, and Daniel P. W. Ellis, "mir_eval: A Transparent Implementation of Common MIR Metrics", Proceedings of the 15th International Conference on Music Information Retrieval, 2014.

4. Rafii, Z., et al. MUSDB18 - a Corpus for Music Separation. 1.0.0, Zenodo, 17 Dec. 2017, doi:10.5281/zenodo.1117372.